Mass-Spring Damper Array as a Mechanical Medium for Computation*

Yuki Yamanaka[1][0000-0001-7966-8879], Takaharu Yaguchi[2][0000-0001-9025-6015], Kohei Nakajima[3][0000-0001-5589-4054] and Helmut Hauser[4][0000-0001-5634-7298],

[1] Kobe University, Kobe, Hyogo, Japan
y-yamanaka@stu.kobe-u.ac.jp
[2] Kobe University, Kobe, Hyogo, Japan and JST PRESTO, Kawaguchi, Saitama, Japan
yaguchi@pearl.kobe-u.ac.jp
[3] The University of Tokyo, Bunkyo-ku, Tokyo, Japan and JST PRESTO, Kawaguchi, Saitama, Japan
k_nakajima@mech.t.u-tokyo.ac.jp
[4] University of Bristol, Bristol, UK
helmut.hauser@bristol.ac.uk

Abstract. Recently, it has been reported that the dynamics of mechanical structures can be used as a computational resource—also referred to as morphological computation. In particular soft materials have been shown to have the potential to be used for time series forecasting. Although most soft materials can be modeled by mass-spring systems, a limited number of researches has been performed on the computational capabilities of such systems. In this paper, we propose an array of masses linked in a grid-like structure by spring-damper connections to investigate systematically the influence of structural (size) and dynamic (stiffness, damping) parameters on the computational capabilities for time series forecasting. In addition, such a structure gives us a good approximation of two-dimensional elastic media, e.g., a rubber sheet, and therefore a direct pathway to potentially implement results in a real system. In particular, we compared the mass-spring array to echo state networks, which are standard machine learning techniques for this kind of problems and are also closely related to the underlying theoretical models applied when exploiting mechanical structures for computation. Our results suggest a clear connection of morphological features to computational capabilities.

Keywords: Soft robotics · Morphological computation · Reservoir computing · Mass-spring system · Recurrent neural network.

* Supported by JST, PRESTO Grant Number JPMJPR15E7 and JPMJPR16EC, Japan and by the Leverhulme Trust Research Project Grant RPG-2016-345.

1 Introduction

In recent years, a new field of robotics, called Soft Robotics, has been risen, see [12]. It uses materials and actuation systems that go beyond conventional building blocks, i.e. rigid body parts and electric motors. This includes a wide range of new, soft materials like silicone, electro-active polymers, gels, and many others, see, e.g. [22]. Despite its success, the field is still struggling to find corresponding control approaches that work with the highly nonlinear dynamics of these materials. One possibility could be to use these, otherwise unwanted morphological features, for our advantage. Instead of controlling every single degree of freedom, we could exploit the underlying complex dynamics as a computational resource. This is often referred to as Morphological Computation, see [20, 21]. Hauser et al. demonstrated with the help of randomly connected networks of nonlinear mass-spring dampers that such dynamics can be indeed used as a computational resource, see [5] and [6]. The underlying theoretical framework is provided by a machine learning technique called reservoir computing [9,13,14,24]. It uses a high-dimensional nonlinear dynamical system, i.e. the *reservoir*, as a computational resource by exploiting it as a temporal kernel in the machine learning sense. Only the weights in the output layer are trained while the structure of the reservoir is typically randomly initialized and then fixed. Interestingly, the reservoir can be implemented by a wide range of dynamical systems leading to various types of reservoir computing. Typical examples from simulations include the echo state network [7, 9] and the liquid state machine [14]. Moreover, even real physical systems can serve as reservoirs as long as they have the necessary properties, see [4]. For example, reservoirs have been built with lasers [19] or even with a bucket of water [3].

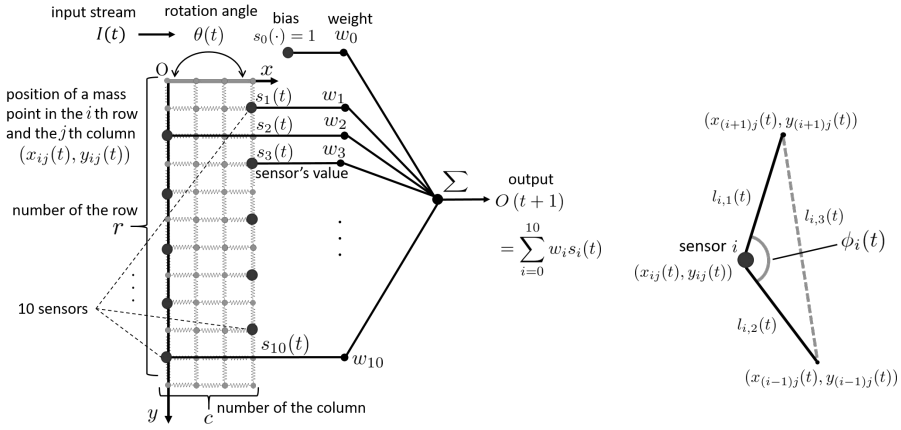
Hauser et al. [5] showed that mechanical structures can be used as reservoirs as well. Interestingly, the mass-spring damper networks they proposed, are also a good approximation of soft structures, e.g. elastic sheets, silicone structures, or even biological tissue. Nakajima et al. used this insight to exploit the dynamics of an octopus-inspired arm, which was modeled by a mass-spring array, as a computational resource [11, 16, 25, 26]. In addition, they showed that this approach is also transferable to real platforms. They used platforms by using an octopus-inspired robot arm build out of silicone to carry out computation and even control [15, 17, 18]. The same approach has been applied also to other robotics platforms, e.g., in locomotion [27] and in trajectory control of a pneumatic arm [2]. However, in the theoretical frameworks as well the implementations in simulation and real robot platforms, the morphological structures are typically fixed. Nevertheless, it has been speculated that there is a clear connection between the morphological features and the computational capabilities of the reservoir, see [4]. Urbain et al. recently performed studies on the trade-offs between morphology, efficiency of control and the ability as a computational resource [23]; however, so far, to the best of the authors' knowledge, there has been very little work done on systematically investigating of how morphological features (like size and form of the network and dynamic properties like stiffness and damping) have influence on the computational performance.

Therefore, in this paper, we propose a structured mass-spring damper array to investigate this question systematically. As computational benchmark tasks we use the approximation of various nonlinear auto regression moving average models (NARMA models) as proposed and used previously by [17]. Furthermore, we compare the results to a standard echo state network, which is a standard tool in machine learning for these kind of tasks.

2 Mass-Spring Damper Array

In this paper, we employed a simulated mass-spring damper array, see Fig. 1(a), as a reservoir. The use of this mass-spring damper array is motivated by the device that was introduced in [17]. They used a silicone based arm inspired by an octopus arm. They added bending sensors and attached it to a motor to actuate the otherwise passive arm. Using this device, they showed that soft structure can be used as a computational resource.

The motor was located on the top of the body and served as an input device. The 5 sensors on each side functioned as outputs. In this paper, we use a similar device by using the model shown in Fig. 2. The body made of the soft material is modeled by the 2-dimensional mass-spring damper grid, which consists of $r \times c$ mass points. The position of a mass point in the ith row and the jth column is defined as (x_{ij}, y_{ij}). The



(a) (b)

Fig. 1. (a) is an illustration of the mass-spring damper array. (b) describes the ith sensor node. The output $s_i(t)$ of the sensor is expressed as follows: when two springs vertically connected to the node are on a straight line (in this case $\phi_i(t) = \frac{\pi}{2}$), $s_i(t) = 0$; when these bend outside of the mass-spring damper array, $\phi_i(t)$ and $s_i(t)$ take positive values ($s_i(t) = 1$ when $\phi_i(t) = \frac{\pi}{4}$); when these bend inside, $\phi_i(t)$ and $s_i(t)$ take negative values ($s_i(t) = -1$ when $\phi_i(t) = -\frac{\pi}{4}$).

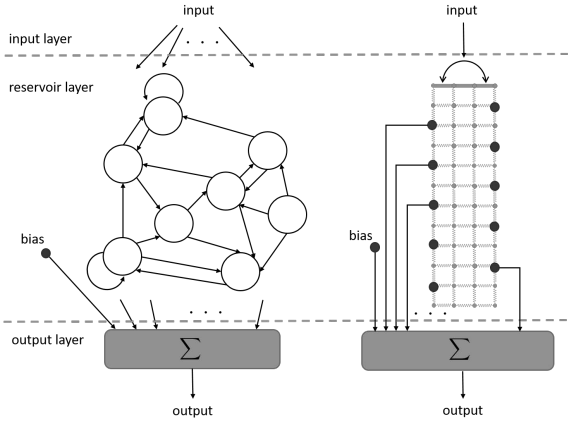


Fig. 2. Correspondence between the mass-spring damper system and echo state networks

ith sensor output at time t is denoted by $s_i(t)$, where s_0 is assumed to be the bias; $s_0(t) = 1$ for all t. Each $s_i(t)$ is computed from the angle $\phi_i(t)$ between the two springs that are connected to ith sensor in the vertical direction. This angle is obtained by

$$\phi_i(t) = \pm \arccos\left(\frac{l_{i,1}(t)^2 + l_{i,2}(t)^2 - l_{i,3}(t)^2}{2l_{i,1}(t)l_{i,2}(t)}\right) \tag{1}$$

where $l_{i,1}$, $l_{i,2}$ respectively denote the distances from the sensor to the two neighboring mass points in the vertical direction, and $l_{i,3}$ is that between the two neighboring points. The sign of ϕ_i is determined by the positions of the sensor and the two neighboring mass points. Each output $s_i(t)$ is defined as $s_i(t) = \frac{4}{\pi}\phi_i(t)$ to be the normalized value of $\phi_i(t)$ so that $|s_i(t)| = 1$, if $\phi_i(t) = \pm\frac{\pi}{4}$(see Fig. 1(b)). The output $O_{\mathrm{MS}}(t+1)$ is the weighted sum of the outputs of the sensors

$$O_{\mathrm{MS}}(t+1) = \sum_{i=0}^{10} w_i^{\mathrm{MS}} s_i(t), \tag{2}$$

where $\mathbf{W}^{\mathrm{MS}} = [w_0^{\mathrm{MS}}, \ldots, w_{10}^{\mathrm{MS}}]^\top$ are the weights. The superscript MS on the symbols is the abbreviation of "Mass-Spring." This superscript is used for distinction of these symbols from those by echo state networks, which are introduced in Section 4.

We assume that the mass-spring array is under the effect of gravity, which acts in positive y direction. We also assume that a damping force exists between each neighbouring pair of the mass points, and that the mass points $(x_{1j}, y_{1j})(j = 1, \ldots, c)$ are fixed on the top of the device on line by rigid horizontal connections. For the sake of simplicity the masses of the all mass points are assumed to be a same value $m = 1.0$. The springs are also assumed to be uniform, i.e., they all have the same spring constant

k and the same equilibrium length l_s and the same damping coefficient γ . Under these assumptions the equation of motion of each x_{ij} is derived in a straightforward way as

$$k\Bigg\{ -4x_{ij} + x_{(i+1)j} + x_{(i-1)j} + x_{i(j+1)} + x_{i(j-1)}$$

$$+ l \ \frac{x_{ij}-x_{(i-1)j}}{\sqrt{(x_{ij}-x_{(i-1)j})^2+(y_{ij}-y_{(i-1)j})^2}} + \frac{x_{ij}-x_{(i+1)j}}{\sqrt{(x_{ij}-x_{(i+1)j})^2+(y_{ij}-y_{(i+1)j})^2}}$$

$$+ \frac{x_{ij}-x_{i(j-1)}}{\sqrt{(x_{ij}-x_{i(j-1)})^2+(y_{ij}-y_{i(j-1)})^2}} + \frac{x_{ij}-x_{i(j+1)}}{\sqrt{(x_{ij}-x_{i(j+1)})^2+(y_{ij}-y_{i(j+1)})^2}}\Bigg)\Bigg\} - m\ddot{x}_{ij} = \gamma\dot{x}_{ij} \tag{3}$$

and

$$k\Bigg\{ -4y_{ij} + y_{(i+1)j} + y_{(i-1)j} + y_{i(j+1)} + y_{i(j-1)}$$

$$+ l \ \frac{y_{ij}-y_{(i-1)j}}{\sqrt{(x_{ij}-x_{(i-1)j})^2+(y_{ij}-y_{(i-1)j})^2}} + \frac{y_{ij}-y_{(i+1)j}}{\sqrt{(x_{ij}-x_{(i+1)j})^2+(y_{ij}-y_{(i+1)j})^2}}$$

$$+ \frac{y_{ij}-y_{i(j-1)}}{\sqrt{(x_{ij}-x_{i(j-1)})^2+(y_{ij}-y_{i(j-1)})^2}} + \frac{y_{ij}-y_{i(j+1)}}{\sqrt{(x_{ij}-x_{i(j+1)})^2+(y_{ij}-y_{i(j+1)})^2}}\Bigg)\Bigg\} - m\ddot{y}_{ij} + mg = \gamma\dot{y}_{ij}. \tag{4}$$

3 Approximation of NARMA Models as a Benchmark Test

In order to illustrate the potential effectiveness of the mass-spring damper array as a mechanical medium for computation, we performed the following tests. We used the problem of approximation of outputs of NARMA(n) models for $n = 2, 10, 20$ as benchmarks:

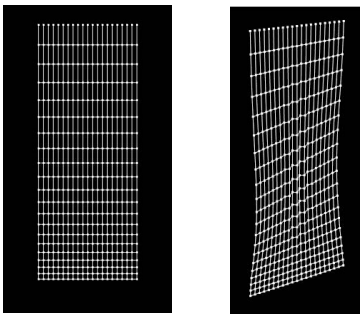
NARMA2

$$y(t+1) = \begin{cases} 0.4y(t) + 0.4y(t)y(t-1) + 0.6I^3(t) + 0.1 & (t \geq 0) \\ 0 & (t \leq -1) \end{cases} \tag{5}$$

NARMA10,20

$$y(t+1) =$$

$$\begin{cases} 0.3y(t) + 0.05y(t)\left(\sum_{j=0}^{n-1} y(t-j)\right) + 1.5I(t-n+1)I(t) + 0.1 & (t \geq 0) \\ 0 & (t \leq -1) \end{cases} \tag{6}$$



(a) (b)

Fig. 3. Visualization of a mass-spring network using WebGL. (a) is a network under the force of gravity. Note that connections on the top are more stretched as they carry more weight than connections at the bottom. (b) is a snapshot of mass-spring system with the top rotated. The input $I(t)$ is applied to the network as the rotational angle of the top (see [17] for details).

where $n = 10, 20$ respectively for NARMA10, 20. These models have been proposed and used by [1] for the evaluation of recurrent neural networks. However, they have been used as a benchmark in various other studies, e.g., [5, 8, 17, 24].

Following [17], we used

$$I(t) = \begin{cases} 0.2 \sin\left(2\pi f_1 \frac{t}{T}\right) \sin\left(2\pi f_2 \frac{t}{T}\right) \sin\left(2\pi f_3 \frac{t}{T}\right) & (t \geq 0) \\ 0 & (t \leq -1) \end{cases} \tag{7}$$

with $(f_1, f_2, f_3) = (2.11, 3.73, 4.33)$ as the input sequence. The parameter T controls the rate of change of $I(t)$; actually $I(t)$ is applied to the mass-spring network as the rotation angle of the top (see Figs. 1 and 2). We set $T = 400$ in this paper.

The model equations (3) and (4) are numerically solved by using the 4th order Runge–Kutta method with a step size of $\Delta t = 0.005$. The computational results are visualized by using WebGL, which is an implementation of OpenGL for Java Script, as shown in Fig. 3. Fig. 3(a) shows the equilibrium state of the mass-spring array only under the influence of gravity and Fig. 3(b) a snapshot of the typical motion of the array introduced by the input and the effect of gravity. The weights $\mathbf{W}^{\mathrm{MS}} = [w_0^{\mathrm{MS}}, \ldots, w_{10}^{\mathrm{MS}}]^\top$ are determined by minimizing the normalized mean square error with $y(t+1)$ for $1 \leq t \leq 5000$ as the training data. More precisely, first we performed numerical simulations until the mass-spring systems reach an equilibrium state (see Fig 3(a)) and we set $t = 0$ at this time. Then the weights are determined by minimizing

$$E = \frac{\sum_{t=1}^{5000} (y(t+1) - O^{\mathrm{MS}}(t+1))^2}{\sum_{t=1}^{5000} y^2(t+1)} \tag{8}$$

so that the squared error between the output of the system and the NARMA models is minimized. As the output is defined by (2), $\mathbf{W}^{\mathrm{MS}}$ is obtained by

$$\mathbf{W}^{\mathrm{MS}} = S^{+}\mathbf{y} \tag{9}$$

where S is the 5000×11 matrix of which row vectors are $s_0(t), \ldots, s_{10}(t)$ for each $t = 1, \ldots, 5000$, S^{+} is the Moore–Penrose pseudoinverse of S and $\mathbf{y} = [y(2), \ldots, y(5001)]^{\top}$.

4 Pretests with Echo State Networks

To get a better understanding of the computational performance of the proposed mechanical structure we compare it to echo state networks (ESNs), which are standard tools to learn dynamical systems like the chosen NARMA tasks. The ESNs will serve as a baseline for comparison. We performed pretests to determine the appropriate values of the parameters for the ESNs. In what follows, results are evaluated by the normalized error

$$E = \frac{\sum_{t=5001}^{10000}(y(t+1)-O(t+1))^2}{\sum_{t=5001}^{10000} y^2(t+1)}. \tag{10}$$

Fig. 4 is a schematic description of ESNs. We denote the numbers of input nodes, internal nodes and output nodes by K, N and L respectively. We also denote the $N \times (1+K)$ weight matrix from the input layer to the reservoir layer by $W^{\mathrm{in}} = [w_{ij}^{\mathrm{in}}]$, the weights between the nodes in the reservoir by $N \times N$ matrix $W = [w_{ij}^{\mathrm{in}}]$, and the $L \times (1+K+N)$ weight matrix from the reservoir layer to the output layer by $W^{\mathrm{out}} = [w_{ij}^{\mathrm{out}}]$. The bias is denoted by b. We used leaky integrator echo state networks because the input sequence (7) has low frequency modes and leaky integrator ESNs are suitable for such sequences (see [10]). The output of leaky integrator ESNs is denoted by $O_{\mathrm{ESN}}(t)$ in the following update equations:

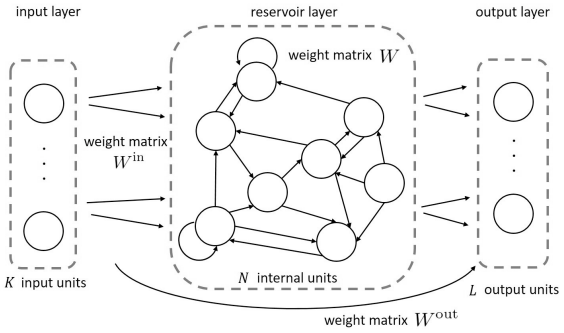


Fig. 4. Echo state network setup. The new input to the internal nodes of the reservoir layer caused by propagation from the input layer and the reservoir layer itself is represented by $\tilde{x}(t)$. $\tilde{x}(t)$ flows into the internal nodes at the leaking rate α, and then values of the internal nodes $x(t)$ is obtained.

$$\tilde{x}(t) = \tanh\left(W^{\text{in}}[b, I(t)]^\top + Wx(t-1)\right), \tag{11}$$
$$x(t) = (1-\alpha)x(t-1) + \alpha\tilde{x}(t), \tag{12}$$
$$O_{\text{ESN}}(t) = W^{\text{out}}[b, I(t), x(t)]^\top, \tag{13}$$

where the vector $x(t)$ represents the values of the internal nodes and α is the leaking rate, which controls the speed of dynamics, and is fixed to 0.3 for simplicity. We use the hyperbolic tangent function as the activation function, and set $K = 1$ and $L = 1$. Each component of W^{in} is randomly set to one of the three values of 1.0, -1.0, 0 with probabilities 2.5%, 2.5%, 95% respectively. Similarly, the weights in W are set to w, $-w$ or 0 with probabilities 2.5%, 2.5%, 95% and with a fixed $w > 0$.

We performed the benchmark tests, changing the size N of the reservoir (100 or 200 nodes) and the weights. For each choice of the parameters N and w, 20 reservoirs were randomly generated.

The performance of ESNs is dependent on the spectral radius[5] ρ of the matrix W, see [7]. The objective of the first test is investigation of the actual dependence of the performance on the spectral radius. The results of the approximation tests of NARMA2, 10, 20 are shown in Fig. 5. The horizontal axis shows the spectral radius ρ of the weight matrix W, and the vertical axis the normalized squared error. Note that the spectral radius becomes larger as each of N and w takes a larger value.

Fig. 5(a),(b),(c) show the results for NARMA2, 10, 20. In these figures, no significant difference in the dependence of the accuracy on the spectral radius ρ is observed. In all of the figures, the performance of the ESNs with $(N, w) = (200, 2.0)$, $(100, 4.0)$, $(200, 4.0)$ was stable in the sense that the accuracy with these parameters was almost the same among the 20 trials. Meanwhile, the ESNs with $(N, w) = (100, 0.4)$, $(200, 0.4)$, $(200, 0.5)$, $(200, 1.0)$, $(100, 2.0)$ often show a worse performance. In particular, the deviations of the errors by the networks with $(N, w) = (100, 0.4)$, $(200, 0.4)$, $(200, 0.5)$ are quite large.

The ESN with the best accuracy was obtained when the parameters were set to $(N, w) = (200, 0.4)$, which gave $\rho \simeq 1.32$. The errors in this case were about 10^{-8} for the NARMA2 test, and about 10^{-7} for NARMA10 and NARMA20, while when in most cases with $\rho > 3$ the errors were around 10^{-5} for NARMA2, 10^{-2} for NARMA10 and 10^{-3} for NARMA20. This dependence of the performances on ρ may be due to whether the generated reservoir had the echo state property or not, see [7].

Despite the fact that approximation of NARMA models with higher degree is known to be a difficult task, the accuracy of the networks with $\rho > 3$ was better for NARMA20 than for NARMA10. This implies that networks with a large ρ have a different property from standard ESNs with a small ρ. When the spectral radius is small

[5] The spectral radius of the matrix is the largest absolute value of the eigenvalues of the matrix. The performance of ESNs strongly depends on if the network has the so-called echo state property, and it is known that the small spectral radius indicates this property. See [7] for detail.

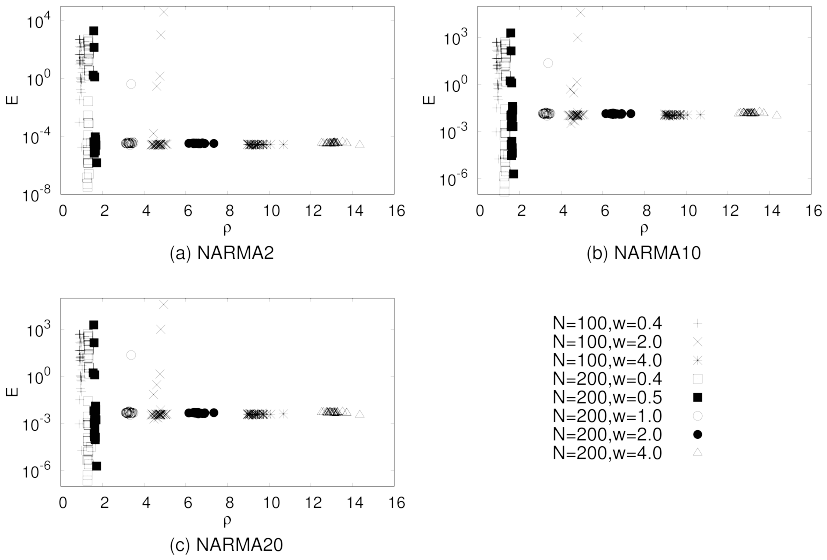


Fig. 5. Results of echo state networks with various N and w

($\rho < 2$), the deviation of the order of accuracy was quite large. That for the results for NARMA2, for example, was from 10^{-8} to 10^{3}. Similar results are also reported in [17].

5 Results of the Benchmark Tests of the Mass-Spring Damper Array

First, we compared performances of the mass-spring arrays with averaged errors of 20 randomly generated ESNs with $(N, w) = (200, 4.0)$, which gave the best results in the previous tests. Table 1 shows the averaged errors over all experiments of the mass-spring array performed with various physical parameters and the averaged and the smallest errors of the ESNs with the above parameters. As illustration, we also show in Fig. 6 the input signal and examples of the outputs of the array with the parameters $(r, c, m, l, k, \gamma) = (50, 50, 1.0, 1.0, 3000, 0.05)$ for $5001 \leq t \leq 5500$, along with the results by the ESN with $(N, w) = (200, 4.0)$.

Table 1. The averaged errors by the mass-spring damper array (MS) and by the echo state networks (ESN) along with the best results of echo state networks

	average of MS	average of ESN	best of ESN
NARMA2	3.93×10^{-5}	3.31×10^{-5}	3.34×10^{-8}
NARMA10	2.65×10^{-3}	1.43×10^{-2}	1.52×10^{-7}
NARMA20	1.93×10^{-3}	4.91×10^{-3}	2.36×10^{-7}

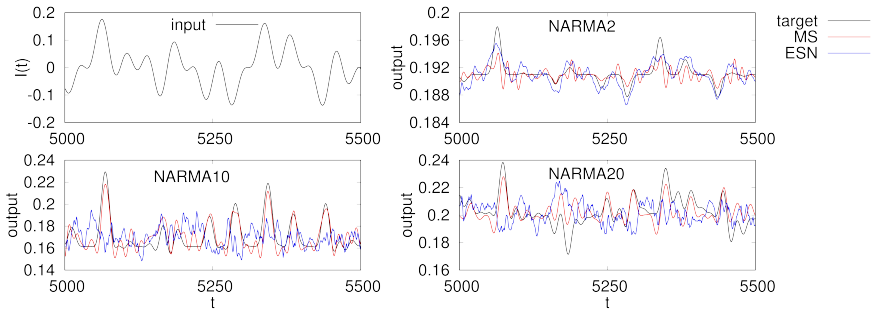


Fig. 6. Examples of the results of the NARMA tasks. In the legend, "target" corresponds to the output of the NARMA model, "MS" to that of the mass- spring array and "ESN" to that of the echo state network.

Next, we investigated relations between parameters of the mechanical structure (i.e., the size and the dynamical parameters as well) and the performance of the system through some tests.

Firstly, we observed the dependence of the performance on the size of the mass-spring damper system by performing the tests with various r and c; $r = 10, 20, \ldots, 100, c = 10, 20, \ldots, 100$. The results are shown in Fig. 7(a), (b) and (c). These figures show that there exists a certain dependence between the size of the array and the accuracy of the mass-spring damper system. For example, in the results of the tests for NARMA2 shown in Fig. 7(a), the parameters $r = 40, c = 80, 90, 100$ and $r = 100, c = 90, 100$ gave better results than others. The best accuracy was achieved when $r = 100, c = 100$, and in that case, the error E was about 0.000027. Interestingly there exist two local optima around $r = 40, c = 80, 90, 100$ and $r = 100, c = 90, 100$. A remarkable conclusion is that outputs of larger systems, which have a larger number of degrees of freedom, are not always more accurate than smaller systems.

Secondly, similarly we investigated the dependence of the performance on the spring constant k and the damping coefficient γ. In the tests we tried various values of k and γ with r, c, m, l fixed to $r = 50, c = 50, m = 1.0, l = 1.0$. The results for $k = 500, 1000, ..., 10000$ and $\gamma = 0.01, 0.02, ..., 0.2$ are shown in Fig. 7(d), (e) and (f). In the results of the NARMA2 test shown in Fig. 7(d), larger γ gives higher accuracy, while the best k was around 3500. Fig. 7(e) and (f) show the results for the NARMA10 and NARMA20 tasks.

It is clearly shown in Fig. 7(d), (e) and (f) that smaller k is suitable for these tasks. In particular, optimal values of k for NARMA10 and NARMA20 tasks are possibly less than 500, which is the smallest value of k plotted in Fig. 7(e) and (f). Therefore we performed the additional tests using $k = 50, 100, ..., 500$, of which results are shown in Fig. 7(g) and (h).

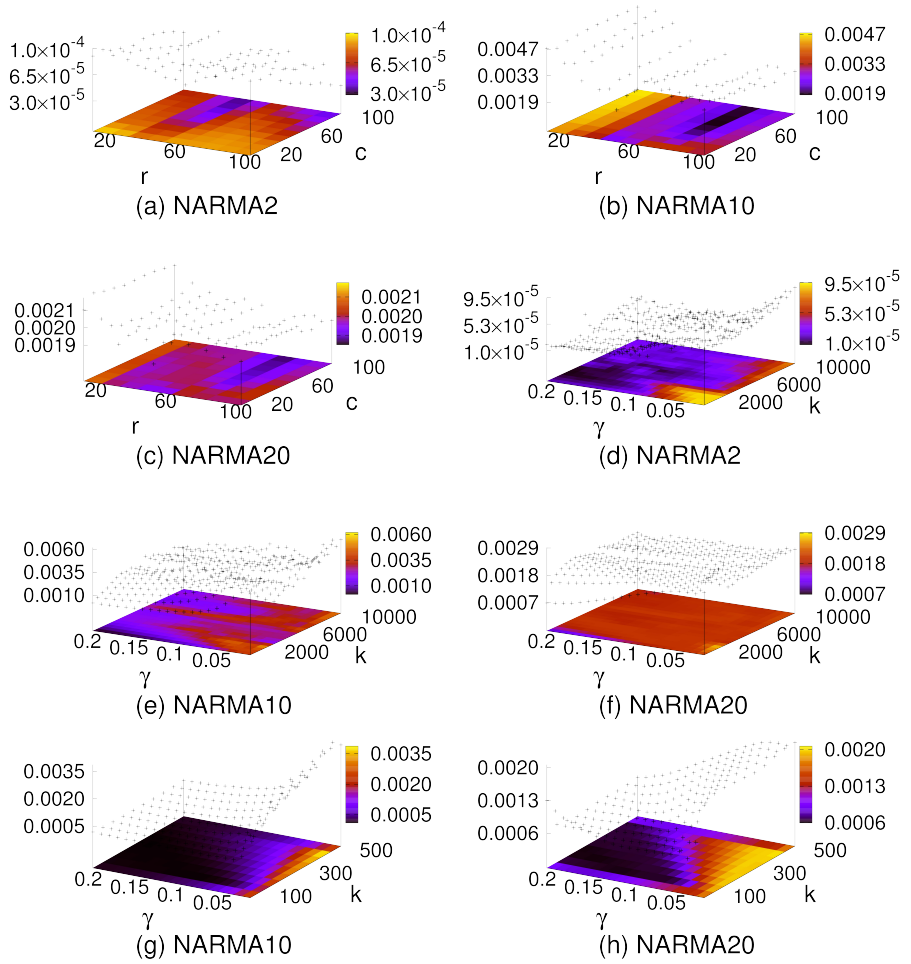


Fig. 7. (a), (b) and (c) are the results by the mass-spring damper array with various r and c. The vertical axis corresponds to the normalized mean squared error between the outputs of the mass-spring damper array and those of the NARMA models. (d), (e) and (f) are the results by the mass-spring damper array with various k and γ. (g) and (h) are enlarged graphs of the results of the mass-spring damper array for smaller k.

6 Discussion

The results presented in Fig. 7(a), (b) and (c) suggest an interesting conclusion. The pure number of mass points is not enough to determinate the performances of the system. For example, the accuracy of the system with $r = 40, c = 100$ was better than that of the system with $r = 100, c = 40$. This implies that the performance of the system

depends in some sense on the two dimensional shape of the medium rather than just the size. This is in so far interesting as the theoretical models proposed by Hauser et al. predict that the higher dimensional the reservoir is the more likely the computational power would increases. The difference here could be mainly due to the existence of the gravity force. Because of the gravity force, the motion of the system is larger in the y direction than in the x direction, and hence changes of c, which is the number of the mass points in the y axis, are more affected by the motion of the system. In addition, we have artificially introduced asymmetry in the structure by allowing sensors only on the side and input on the top.

The difference of the dependence of the performance on r and c was also observed in the results in NARMA10 and NARMA20, which are shown in Fig. 7(b), (c). In these tests the systems with $r = 70, 80, c = 60, 70, 80, 90, 100$ yield better results than the others for both the NARMA10 and the NARMA20 tasks. It should be noted that the optimal parameters for NARMA2 are different from those of NARMA10 and NARMA20. This confirms that when the mass-spring array is used as a mechanical medium for computation, morphological parameters related to the size or the shape of the array must be carefully chosen when considering a specific computational task.

Regarding the tests where various k and γ are investigated, for all of the NARMA2, 10, 20 tests, the performance depended on both parameters. In particular, the results in Fig. 7(g), (h) show that outputs of systems with larger γ are more accurate than those with smaller γ, meanwhile systems with smaller k yield more accurate results than those with larger k. Moreover, the values of the error in Fig. 7(g), (h) are smoothly dependent on k and γ, thereby implying existence of a simple function that relates the error function to k and γ.

Because γ is the damping coefficient, the motion of the mass-spring damper system can become unstable when γ is set to a small value. In contrast, when γ is large, the whole systems tends to act as one rigid block. Similarly, larger k makes the motion of the system more dynamic (i.e., can move at higher frequencies), and motions of systems with smaller k are softer (i.e., moves slowly). Hence, for NARMA(n) tasks with higher n mass-spring damper arrays with stiffer behavior (i.e., move rigidly (larger γ) and slowly (smaller k)) are seemingly more suitable than the systems with more dynamic behavior.

In the systems with more active motions, a motion of a mass point wields a great influence on the neighbouring points. In comparison with ESNs, if the mass-spring damper array is in some sense equivalent to the reservoir of an ESN, links between the mass points with greater interaction may correspond to strong links between the neighbouring nodes in the reservoir of the ESN, thereby possibly corresponding to the weight matrix with large weights. Similarly the mass-spring systems with stiffer behavior should correspond to reservoirs with small weights. In the previous tests of ESNs, it was shown that if the spectral radius of the weight matrix is large, the accuracy of outputs of the systems is feasible, but not excellent, while the deviations of the errors are small. The spectral radius is dependent on the size and the weights of the reservoir; in particular large weights yield a large spectral radius. Hence, it is expected that deviations of the errors

with large k and small γ must be small, and the outputs are not extremely accurate. The results in Fig. 7(f) and Fig. 7(g), (h) are compatible with this expectation.

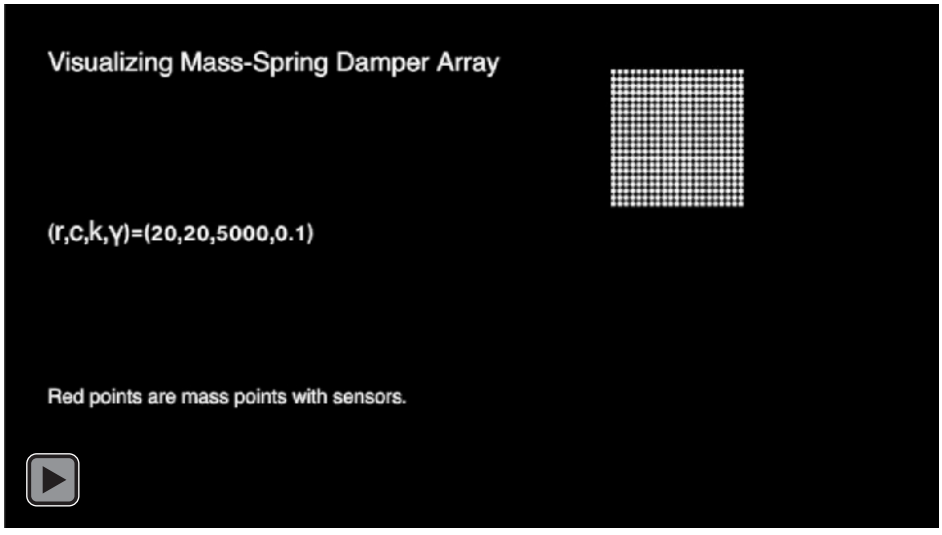
In this paper, we used the product of three sinusoidal signals as the input, different trends may appear when other input time series are imposed (e.g., random input time series from uniform distribution). These cases with other benchmarks will be investigated in future work.

7 Conclusion

In this paper we have investigated by simulations the performance of the mass-spring damper array as a computational medium, in particular, to what extent the morphological parameters affect the overall computational performance.

The errors of the mass-spring damper array for the NARMA tests were almost same as the averaged errors of the ESNs, but still larger than for the best perform ESNs. However, considering that the deviation of performances of ESNs with small spectral radius was very large, the stable performance of the mass-spring damper system was remarkable. This implies that when the proposed mechanical array can be used as a medium for computation, it should surely work to a certain extent although the accuracy may not be excellent. This stability of the performance of the mass-spring array would be advantageous for real applications in terms of ease-of-use without strict tuning parameters.

Supplementary material



References

1. Atiya, A.F., Parlos, A.G.: New results on recurrent network training: unifying the algorithms and accelerating convergence. IEEE transactions on Neural Networks **11**(3), 697–709 (2000)
2. Eder, M., Hisch, F., Hauser, H.: Morphological computation-based control of a modular, pneumatically driven, soft robotic arm. Advanced Robotics **32** (7), 375–385 (2018). https://doi.org/10.1080/01691864.2017.1402703
3. Fernando, C., Sojakka, S.: Pattern recognition in a bucket. In: European Conference on Artificial Life. pp. 588–597. Springer (2003)
4. Hauser, H., Füchslin, R., Nakajima, K.: Morphological computation—the physical body as a computational resource. In: Opinions and Outlooks on Morphological Computation, editors Hauser, H.; Füchslin, R.M. and Pfeifer, R., Chapter 20, 226–244, ISBN 978-3-033-04515-6 (2014)
5. Hauser, H., Ijspeert, A.J., Füchslin, R.M., Pfeifer, R., Maass, W.: Towards a theoretical foundation for morphological computation with compliant bodies. Biological Cybernetics **105**(5), 355–370 (2011)
6. Hauser, H., Ijspeert, A.J., Füchslin, R.M., Pfeifer, R., Maass, W.: The role of feedback in morphological computation with compliant bodies. Biological Cybernetics **106**(10), 595–613 (2012). https://doi.org/10.1007/s00422-012-0516-4
7. Jaeger, H.: Adaptive nonlinear system identification with echo state networks. In: Advances in Neural Information Processing Systems. pp. 609–616 (2003)
8. Jaeger, H.: Adaptive nonlinear system identification with echo state networks. In:Advances in Neural Information Processing Systems. pp. 609–616 (2003)
9. Jaeger, H., Haas, H.: Harnessing nonlinearity: Predicting chaotic systems and saving energy in wireless communication. Science **304**(5667), 78–80 (2004)
10. Jaeger, H., Lukoševičius, M., Popovici, D., Siewert, U.: Optimization and applications of echo state networks with leaky-integrator neurons. Neural Networks **20**(3), 335–352 (2007)
11. Kang, R., Kazakidi, A., Guglielmino, E., Branson, D.T., Tsakiris, D.P., Ekaterinaris, J.A., Caldwell, D.G.: Dynamic model of a hyper-redundant, octopus- like manipulator for underwater applications. In: 2011 IEEE/RSJ International Conference on Intelligent Robots and Systems. pp. 4054–4059 (2011). https://doi.org/10.1109/IROS.2011.6094468
12. Laschi, C., Mazzolai, B., Cianchetti, M.: Soft robotics: Technologies and systems pushing the boundaries of robot abilities. Sci. Robot. **1**(1), eaah3690 (2016)
13. Lukoševičius, M., Jaeger, H.: Reservoir computing approaches to recurrent neural network training. Computer Science Review **3**(3), 127–149 (2009)
14. Maass, W., Natschläger, T., Markram, H.: Real-time computing without stable states: A new framework for neural computation based on perturbations. Neural Computation **14**(11), 2531–2560 (2002)
15. Nakajima, K., Li, T., Hauser, H., Pfeifer, R.: Exploiting short-term memory in soft body dynamics as a computational resource. Journal of The Royal Society Interface **11**(100) (2014)
16. Nakajima, K., Hauser, H., Kang, R., Guglielmino, E., Caldwell, D., Pfeifer, R.: A soft body as a reservoir: case studies in a dynamic model of octopus-inspired soft

robotic arm. Frontiers in Computational Neuroscience **7**, 91 (2013). https://doi.org/10.3389/fncom.2013.00091
17. Nakajima, K., Hauser, H., Li, T., Pfeifer, R.: Information processing via physical soft body. Scientific Reports **5** (2015)
18. Nakajima, K., Hauser, H., Li, T., Pfeifer, R., Exploiting the Dynamics of Soft Materials for Machine Learning. Soft Robotics **5** (3), 339–347 (2018)
19. Paquot, Y., Duport, F., Smerieri, A., Dambre, J., Schrauwen, B., Haelterman, M., Massar, S.: Optoelectronic reservoir computing. Scientific Reports **2**, 287 (2012)
20. Paul, C., Valero-Cuevas, F.J., Lipson, H.: Design and control of tensegrity robots for locomotion. IEEE Transactions on Robotics **22**(5), 944–957 (2006)
21. Pfeifer, R., Gómez, G.: Morphological computation–connecting brain, body, and environment. In: Creating brain-like intelligence, pp. 66–83. Springer (2009)
22. Rus, D., Tolley, M. T.: Design, fabrication and control of soft robots. Nature **521**(7553), 467–475 (2015)
23. Urbain, G., Degrave, J., Carette, B., Dambre, J., Wyffels, F.: Morphological properties of mass-spring networks for optimal locomotion learning. Frontiers in Neurorobotics **11**, 16 (2017). https://doi.org/10.3389/fnbot.2017.00016
24. Verstraeten, D., Schrauwen, B., d' Haene, M., Stroobandt, D.: An experimental unification of reservoir computing methods. Neural Networks **20**(3), 391–403 (2007)
25. Yekutieli, Y., Sagiv-Zohar, R., Aharonov, R., Engel, Y., Hochner, B., Flash, T.: Dynamic model of the octopus arm.I. biomechanics of the octopus reaching movement. J. Neurophysiol. **94**, 1443–1458 (2005)
26. Yekutieli, Y., Sagiv-Zohar, R., Aharonov, R., Engel, Y., Hochner, B., Flash, T.: Dynamic model of the octopus arm.II. control of reaching movements. J. Neurophysiol. **94**, 1459–1468 (2005)
27. Zhao, Q., Nakajima, K., Sumioka, H., Hauser, H., Pfeifer, R.: Spine dynamics as a computational resource in spine-driven quadruped locomotion. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2013). pp. 1445–1451. IEEE (2013)